The Security Risks Posed By APIs & How To Mitigate Them

Introduction

In the world of software development today, Application Protocol Interfaces (APIs) are one of the key packages that are being used in creating applications today. But first, what exactly is an API? It can be thought of as an intermediary between two very different software packages which brings them together in order to create a seamless environment for the end user.

In other words, it is the bridge that takes your request from a page on a website that you are visiting back to the web server.

In response, the web server then analyzes the query, calls up the relevant information from the database, and then transmits the results back to you in a manner that is consumable. A good example of this is requesting a free whitepaper from a Cybersecurity site.

You fill in your relevant contact information, and this gets transmitted to the web server. It then searches its own database and relevant directory structures for the whitepaper, and then sends it back to your Email inbox.

Think of the API as the tunnel between that contact form and the whitepaper you have selected. If you decide later on that you want another piece of content resource, that same API can be used over again to call that particular piece up.

The primary advantage of the API is that it can be used over and over again, for differing requests. If it hadn't been for the API, unique source code would have to be implemented each and every time for every different kind of request.

The Security Risks That Are Posed To APIs

The website of the Open Web Application Security Project, also known as OWASP, is one of the best resources to find out what out the latest risks are. This organization updates their list every few years, and in fact, the latest version came out this year. According to them, here are the top API Security Risks:

1) Broken Access Control
2) Cryptographic Failures
3) Injection Style Attacks
4) Insecure Source Code Design
5) Security Misconfiguration
6) Vulnerable/Outdated Components
7) IAM Failures
8) Data Integrity Failures
9) Security Logging/Monitoring Failures
10) Server-Side Request Forgeries

How To Mitigate API Security Vulnerabilities

What can a business do to mitigate some of these security gaps? Here are some top tips:

- Authentication should come first, then authorization:

  There is often confusion as to what these two are. Authentication is confirming the actual identity of the end user, and authorization are the permissions, rights, and privileges that are granted to them. In most organizations, the latter is usually done first, and the former second. But in terms of API security, this way of thinking drastically needs to be changed. Before an end user can be granted access to any shared resource, their identity must first be confirmed, ideally through Multifactor Authentication (aka MFA, this is where more than one layer of authentication is used). Once this process has been accomplished, then the end user should be given the appropriate privilege level to access what he or she needs to. This could fall into the realm of policy-based access control (PBAC), or of role-based access control (RBAC). To ensure an even greater level of security for the APIs, access tokens should also be used.

- Make regular use of Cryptography:

  In this regard, the tools of Encryption must be used to protect the APIs that are being used in the web application, especially when it comes to the point where of communications where the end user is requesting certain pieces of information, and the web server has to respond to them. The primary objective of this is to ensure the highest level of security for any authentication details that are being sent back and forth. Examples of this include implementing SSL certificates, Transport Layer Security (aka TLS) protocols, and API gateways (which will allow you to streamline and manage all of the network traffic coming to and leaving the APIs).

- Deploy Throttling Quotas:

  As described previously in the last section, of the main security weaknesses of APIs is that many of them do not have any sort of restrictions placed on them in terms of the number of requests that they can process. In order to mitigate this risk, it is highly recommended that certain rules be deployed onto the APIs to gradually reduce the number of requests that they can process once a certain limit has been reached. For example, this can include the following:

  *A maximum number of requests that can be handled during certain time periods;

  *Imposing limitations on the level of bandwidth that is being consumed;

  *Deploying other types of quotas that are time or resource based in nature.

- Make use of validation techniques:

  Also as mentioned, of the other grave weaknesses of APIs is that malicious code can be injected into them and be manipulated by the Cyberattacker for their gain. One way to overcome this vulnerability is to implement set of validation rules that acts as a cross check for any new lines of source code (such as input strings or objects) that are implemented. In this way, if anything appears to be anomalous or out of the ordinary, these lines of code can be discarded quickly before an injection attack occurs to the APIs that are being used.

- Deploy RESTful APIs:

  This is a set of rules and standards that have been created in order to make APIs easy to understand and scan on a regular basis so that any security vulnerabilities in them can be detected quickly and efficiently. This is also known as the "RESTful Web Service" and is primarily based upon the Simple Object Access Protocol (aka SOAP). To this end, this kind of API forces the software developer to formulate source code from the very beginning of the Software Development Lifecycle (aka SDLC), rather than waiting at the very end, when time to delivery is of essence.

- Implement Auditing and Logging Tools:

  Given the ever-changing dynamics of the security threat landscape, it is now more paramount than ever to keep examining your log files on a regular basis, especially as it relates to the APIs that are currently in place. Much of this can now be automated through the use of Artificial Intelligence (AI) technology and can detect any kind of erratic activity in just a matter of a few seconds.

Conclusions

Overall, this article has examined what an API is, and the major security issues that are associated with them, through the list provided by OWASP. Solutions were also given in helping to mitigate these risks. The use of APIs will only continue to grow into the future, as web applications become more complex and handle an even larger influx of information and data.

Therefore, it is always important to keep ahead of the curve, by conducting regular security audits on the APIs, through the use of Penetration Testing and Threat Hunting exercises.

Sources

1) https://owasp.org/Top10/